Standardizing Security: The case of threshold cryptography

Ran Canetti
Boston University
Goals for standardization

- Creating agreement on an object: Making the world more efficient
  - Common language
    - (Quarter Pounder vs. Royale with Cheese)
  - Interoperability
    - Electric plugs
    - IETF
  - Modular design
    - Program APIs

- Benchmarking: setting common levels of quality and operation

- Protecting business interests

- Getting people from different backgrounds to brainstorm and agree on what works
Standardizing cryptographic protocols

Complex object:
• Several parties, different concerns → security harder to capture

• Depends on other mechanisms:
  • Networking stack
  • Actual network properties
  • Execution environment

Where to draw the line?
Standardize Threshold Cryptography?

• Seriously? Let’s crawl before we run marathons...
Standardize Threshold Cryptography?

• Seriously? Let’s crawl before we run marathons…

But people are using it in practice, and we’ll have to live with whatever they come up with…
Standardize Threshold Cryptography?

• Seriously? Let’s crawl before we run marathons...

But people are using it in practice, and we’ll have to live with whatever they come up with...

➤ Let’s do it right!
Standardizing Threshold Cryptography: Suggested guidelines

• Concentrate on a small set of primitives (eg. threshold signatures)
  • Do we want to concentrate on specific verification algorithms for interoperability? If so then which ones? (ECDSA? Schorr? BLS? EDDSA?) or leave it open?

• Agree on clear APIs for the primitive:
  • With the calling program (the “user”)
    • When should a signature be generated?
  • With OS utilities and service programs
    • Memory, cache
    • Network (channel assumptions?)

• Agree on a set of security properties
  • Unforgeability (for signatures) Sem. Security (for enc)?
  • Distributional equality with some standardized spec? (and why?)
  • Composability/Modularity?
Standardizing Threshold Cryptography: Suggested guidelines

• Once we agree on these, can have a competition for
  • Algorithms
  • Implementations
  • Proofs of cryptographic security
  • Security analysis of implementation
The UC approach: Specification via an Ideal-Service, with composition
[Universally composable security, C20]

Idea:

• Security of a system is reflected only in its effects on the rest of the external environment.

• Therefore to capture the desired security of system P:
  
• Write an “ideal system” $F$ that captures the desired effects: Functionality and security
  
• The proof of security will assert that P can be made to “looks the same” as F to an external environment.

Note: $F$ need not be realistically implemented. All we care about is its responses to the environment.
The ideal threshold signature functionality

[Given: Identities $S = S_1, ... S_n$ of signatories, access structure $A: \{0,1\}^n \rightarrow \{0,1\}$]

**Keygen:** When receiving input “KeyGen” from a subset $\alpha \subseteq S$ s.t. $A(\alpha) = 1$: obtain from Adv a verification key $VK$ and output $VK$ to all.

**Sign:** When receiving input “Sign m” from* a subset $\alpha \subseteq S$ s.t. $A(\alpha) = 1$: hand $m$ to Adv, obtain a signature string $\sigma$, add $(m, \sigma, 1)$ to local database $DB$ and output $\sigma$ to all.

**Verify:** When receiving input “Verify $(VK, m, \sigma)$” from some party $V$:
- If $(m, \sigma, b) \in DB$ then return $b$ to $V$
- Else if $(m, \sigma', 1) \notin DB$ for any $\sigma'$ then return 0 to $V$ ← unforgeability
- Else hand $(m, \sigma)$ to Adv, obtain $b \in \{0,1\}$, add $(m, \sigma, b)$ to $DB$, and return $b$ to $V$.

**Corrupt:** When Adv asks to corrupt / uncorrupt $S_i$, mark $S_i$ as corrupted/uncorrupted.
- While $S_i$ is corrupted, Adv is allowed to approve signing/keygen instead of $S_i$.
- When asked by $S_i$ if corrupted, answer truthfully.
The ideal threshold signature functionality

Provides:
- Clear API with user protocol
- Clear functionality
- Clear security properties
- Composability. Modularity

Does not provide:
- APIs with OS, network services. (Can be added...)

Can do the same for threshold decryption...
apparently they don't call it a quarter pounder with cheese they get the metric